Grid Resource Allocation Agreement Protocol Working Group

Jon MacLaren, Manchester University
Volker Sander, Forschungszentrum Jülich
Wolfgang Ziegler, Fraunhofer–Institute for Algorithms and Scientific Computing

Document: sched-graap-2.0

Category: Informational June 2002

Advanced Reservations
State of the Art

Status of this Draft

This draft provides information for the grid scheduling community. Distribution of this document is unlimited.

Advanced Reservations: State of the Art

1 Introduction

This is a first go at working out what advanced reservation functionality is available in current scheduling systems. I've currently only got LSF and PBSPro. It would be good for other people to add information about other schedulers. I'd like to see information about Paderborn's CCS, OpenPBS/Maui, Loadleveller, perhaps Sun GridEngine. There are probably others that I don't know about. I don't think there's any merit in adding dead schedulers like NQS and NQE; lets focus on the future.

We could add the local enhanced version of the EASY scheduler (derived from the ancient EASY scheduler for the SP2 at Argonne), available for parallel machines and unix clusters. And there is a scheduler from NEC (calles COSY) which was derived from EASY with the same features more or less but based on C++ and Corba. In addition COSY is able to interact with other COSY-like schedulers for co-allocation of resources. Both are of course no commercial products but research prototypes being used at several sites.

The rows in the table are not definitive. In fact, I'd like to see them expanded upon, and re-categorised as time goes on. I'd also like some rows to undergo mitosis as we proceed and know more about how we want advanced reservation to look. For example, currently there is a single property "Reservation can be negotiated", referring to two-phase commit capability. I fully expect this row to split—several times—into rows representing the many things that we will come to expect from two-phase commit style reservations.

One of the key problems about making comparisons of this kind is the "What is a Reservation?" type question. Such questions can lead to discussions that can use up vast amounts of time, and produce little outcome. This section/appendix is not the place for this discussion – it should only say what "Advanced Reservation" means in PBSPro, LSF, etc.. So, I want to keep base definitions minimal if possible, and represent additional properties as rows in the table.

2 Definitions and Terminology

The following may be acceptable, or may clash with other, existing documents – help, please!

User

A user is simply a client of the software. This could be a human (via any kind of interface: command line, GUI, etc.) or some sort of agent software.

Advanced Reservation

"A set of resources, e.g. number of processors, amount of memory, and possibly disk space, which are controlled by the batch scheduling system, and which are then set aside from normal use by batch jobs, and instead made accessible to a subset of users."

Probably the definition is a bit too specific thus running the risk of being limited, e.g. to batch systems. In the dictionary WG we use the following definition:

The process of requesting various resources for use at a later time.

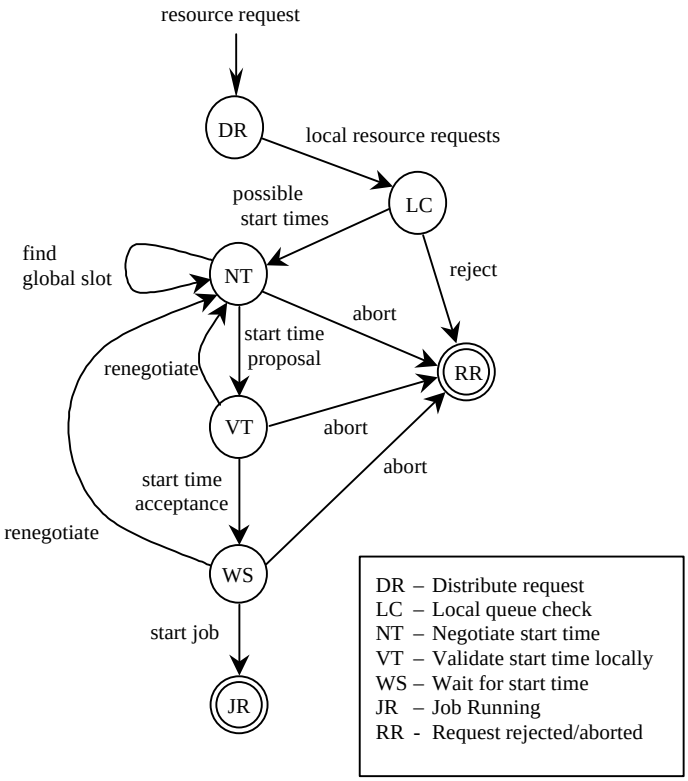
States of Advanced Reservation

All I really wanted here was a list of states that I could talk about later on, when describing properties. But, as I'm writing these, I've started thinking that this should maybe be a state transition diagram. But, if we do that, it probably needs actions on the transitions. That feels like I'm starting to write something like a specification—so for now I won't expand on what's below. (Maybe it should be shorter.)

Requested:	A user has requested a set of resources for a reservation. If the reservation is accepted, it goes to being booked. Otherwise, it becomes declined.
Declined:	The reservation is not successfully allocated for some reason. (Not sure if this state is useful.)
Booked:	A reservation has been made, and will be honoured by the scheduler. From here, the reservation can become active, or be cancelled by the user or system, or be altered.
Booked, change requested:	A user is trying to alter the resources for the reservation prior to its starting. Goes back to booked state on success or failure.
Cancelled:	A user cancels the reservation prior to beginning. Or, the scheduler cancels the reservation prior to it's beginning (this may be due to maintenance downtime being scheduled).
Active:	The reservation has started, but not ended.
Terminated:	A user, or possibly the system, terminates an active reservation before the end-time. (The system may want to do this if the reservation becomes active, but is then idle for a certain length of time.)
Completed:	The reservation continued until it's end-point.

Active, change requested: A user is trying to alter the resources for the reservation after the reservation has become active. Goes back to active state on success or failure.

The list is ok for me but as you say it: the explanations could probaly shorter, i.e. less details.



This is roughly the finite state machine we our meta-scheduler prototype is implementing. We could use this as a starting point if we decide to add something like this to the paper, although I doubt a bit whether the State of the Art appendix is the right place to put it.

3 Properties of Reservations

I've tried to capture properties of reservations below. They're divided into groups or sections. Again, these may be subdivided/moved/changed (as can the groups).

There are other issues with using advanced reservations, like “can/does the scheduler pre-empt to honour a reservation”—these seem to me to be not really germane to the use of reservations, and so I’ve left things like this out.

Think so too, these are local site issues which may change under time and which are in most cases not visible to users or agents quering for a time slots.

3.1 Basic

The scheduler supports Advanced Reservations

We should not document schedulers for which “no” is the answer here. Where the answer is “yes”, I’m assuming that it is possible to cancel a booked reservation, or to abort an active reservation.

Reservations can be made by “normal” users

For some schedulers, reservations can only be made by an administrator. This is not general enough for everyday use of the Grid.

Reservations (can) have their own queue(s)

This is about how work is scheduled within the reservation. Creating a queue to which the users of the reservation can submit jobs seems to be a common approach. If the reservation does have a queue, then jobs can probably be queued to a reservation as soon as the reservation becomes booked.

Interactive jobs can be run in the reservation

If the reservation has been set aside for users, can they do interactive work in this space, i.e. debugging, command line interaction? I’m thinking of work that is submitted immediately to the resource during the time of the reservation. (This allows the users a way of booking time for the debugging large parallel jobs. Even for small parallel jobs, it's much cleaner than competing in a limited set of resources set aside for interactive work.)

Thats the way EASY and COSY work: on submitting a request for resources the users decides whether his job should be run interactive or as a batch job without further intervention. In the later case the user has to submit a script as usual, in the first case he may login to the granted nodes interactively. A scheduler should support both types of usage.

ACL provided for controlling the use of the resources in the reservation

When the reservation is set up, is it possible to allow other users (or groups of users) permission to submit jobs to (or interactively use) the reservation?

ACL(s) provided for the modifying/cancelling/terminating of the reservation

Can the reservation be set up so that users other than the reservation's owner can modify, cancel or terminate the reservation?

3.2 Advanced

A booked reservation be altered

Can the start-time, end-time, number of processors, etc. of a booked reservation be changed after the booking is made?

An active reservation can be altered

Can the resources of the reservation still be altered after a reservation has become active? A particularly interesting example would be a calculation which noted that it was running out of time, and decided to try and extended the reservation by requesting either more processing elements, or more time. (Sounds like something the Cactus team would try to use.)

3.3 Two-Phase Commit

For meta-scheduling on multiple machines to work well, it must be possible for an advanced reservation system to support two-phase commit, where there is negotiation before the reservations are fixed.

(I think that we may need new states for two-phase commit. It may be that "in negotiation" would be a separate state from "requested", or at least a better name.)

A reservation can be negotiated when requested

Does the scheduler permit a user (human or agent) to negotiate the reservation in any way? To differentiate negotiation from what we have now, a Yes/No response to a request does not count.

This will be a large topic. At the moment, no schedulers I'm aware of implement anything here, so it's something that will get expanded in the future.

Negotiation can be used to alter booked reservations

If negotiation were possible when booking, I'd expect the same style of negotiation to be possible when altering booked reservations (if permitted at all).

Negotiation can be used to alter active reservations

If negotiation were possible when booking, I'd expect the same style of negotiation to be possible when altering active reservations (if permitted at all).

4 Scheduling Systems

4.1 LSF

Load Sharing Facility from Platform. Not open source or free.

4.2 PBSPro

Portable Batch System (Professional edition) from Veridian. Not open source or free.

4.3 OpenPBS/Maui

Both PBS and Maui are open source, and free.

4.4 Paderborn CCS

Brief description – help!

4.5 LoadLeveler

Brief description – help! Does this provide advanced reservation without additions?

4.6 LoadLeveler/Maui

Brief description – help!

4.7 Sun GridEngine

Brief description – help! Does this provide advanced reservation?

4.8 EASY (SCAI flavour)

Derived from the Argonne EASY scheduler for the SP2. Ongoing work at the Institute for Algorithms and Scientific Computing SCAI. Supports both batch and interactive jobs. Available and used for several parallel machines and Unix-Clusters. Public domain.

4.9 COSY (NEC)

Derived from the SCAI EASY scheduler. Ongoing work at NEC's C&C Research Laboratory at Sankt Augustin. Supports both batch and interactive jobs. In use at the lab for different PC-Clusters. Binaries available for academic institutions without costs (some kind of contract necessary), source code not available.

5 Capability Table

Taking the properties from Section 3, and the schedulers from Section 4, the following table has been produced. I've only used ticks and crosses. Could use question marks for "don't knows", but I feel this looks too cluttered. As I don't know whether or not LoadLeveler (vanilla) or Sun GridEngine permit reservations, I've not included them in this table.

To save space, I've used **AR** for Advanced Reservation, and **R** for Reservation.

Scheduler ⇒ ⇓ Property	LSF	PBS Pro	Open PBS + Maui	Pader-born CCS	Load Leveler + Maui	EASY	COSY
Supp. AR	√	√	√	√	√	√	√
AR by users	**X**	√				√	√
R has queue	√	√				√	√
R supports interactive jobs	√	√				√	√
ACL for use of R	√	√					
ACL(s) for mod., can. or term. of R							
Can alter booked R							
Can alter active R							
R can be negotiated	**X**	**X**	**X**	**X**	**X**	√	√
Booked R can be re-negotiated	**X**	**X**	**X**	**X**	**X**		
Active R can be re-negotiated	**X**	**X**	**X**	**X**	**X**		

Note that the table probably needs version numbers, if we're not to upset the scheduler developers! That being said, I think this document must only reflect the functionality in existing, released software, i.e. we should not document promises! Special wrappers and add-ons are not widely available, and therefore not very usable in the context of the Grid—we should probably ignore those too...